UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/749,938	12/31/2003	Michael Swafford	50037.0216US01	6670

27488	7590	01/25/2006	

MERCHANT & GOULD (MICROSOFT)
P.O. BOX 2903
MINNEAPOLIS, MN 55402-0903

EXAMINER
IWASHKO, LEV

ART UNIT	PAPER NUMBER
2186	

DATE MAILED: 01/25/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

	Application No.	Applicant(s)
Office Action Summary	10/749,938	SWAFFORD ET AL.
	Examiner	Art Unit	
	Lev I. Iwashko	2186	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *31 December 2003*.

2a)☐ This action is **FINAL.**   2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

     4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *31 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

     Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

     Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

     a)☐ All   b)☐ Some *   c)☐ None of:

       1.☐ Certified copies of the priority documents have been received.

       2.☐ Certified copies of the priority documents have been received in Application No. _____.

       3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

     * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *4/13/2004*.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____ .

DETAILED ACTION

Claim Rejections - 35 USC § 102

1.   The following are quotations of the appropriate paragraphs of 35 U.S.C. 102 that form

the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act

of 1999 (AIPA) and the Intellectual Property and High Technology Technical

Amendments Act of 2002 do not apply when the reference is a U.S. patent resulting

directly or indirectly from an international application filed before November 29, 2000.

Therefore, the prior art date of the reference is determined under 35 U.S.C. 102(e) prior

to the amendment by the AIPA (pre-AIPA 35 U.S.C. 102(e)).

2.   Claims 1-9, 11-19, and 21-29 are rejected under U.S.C. 102(e) as being anticipated by

Abrashkevich et al. (US PGPub 2004/0221120 A1).

Claim 1.   A method for tagging an allocable memory block, comprising: *(Abstract,*

lines 1-3 – State the following: "A data structure, method and system are

provided incorporating a general purpose memory allocator and defensive

heap memory manager)

-       determining the identity of a routine performing one of requesting the

allocable memory block, requesting the size of the allocable memory

block, and freeing the allocable memory block; *(Section 0039, lines 1-*

13 – State the following: "After receiving a request for allocating a

memory pool, the memory manager performs all necessary size

alignments and adjustments related to the specified memory debug

mode (they are described below), allocates a new memory pool from
available free heap space, creates and initializes a pool header
(specific pool header data structures are described below), updates
heap header data using a heap handle provided, and inserts a free
block of the size of total pool space minus pool header size and
memory debug data size into a skip list of free memory blocks. The
heap memory manager returns to the requesting process a pool handle
which in a preferred embodiment contains, but is not limited to, the
following information...)

- generating an identifier for the routine; *(Section 0039, lines 15-19 –*
  Declare a pool identifier)

- and storing the identifier in the allocable memory block. *(Section*
  0041, lines 1-3 – State that the "pool" is stored in an allocable
  memory block)

Claim 2.   The method of claim 1, further comprising examining the heap to
determine the presence of memory errors. *(Section 0029, lines 8-22 –*
Describe how the heap is examined for memory errors)

Claim 3.   The method of claim 1, further comprising performing a checksum on the
allocable memory block and *(Section 0066, lines 18-21 – State that there*
is a checksum done on an allocable memory block)

- storing the results of the checksum within the allocable memory block.
  (Section 0070, lines 15-32 – Show how the checksum is stored in a
  allocable memory block)

Claim 4.   The method of claim 3, further comprising examining the results of the
checksum to determine the presence of memory errors. *(Section 0068,*
lines 1-7 – State that the checksum is examined to see if there are errors)

Claim 5.   The method of claim 1, wherein the identifier is the return address of the
identified routine. *(Section 0039, lines 16-19 – State the following: "a*
pool identifier which in a preferred embodiment is the unique user's pool
offset value, for example an offset from the heap starting address to the

beginning of a pool header without walls and any memory debug attachments")

Claim 6. The method of claim 1, further comprising writing a memory overwrite detection pattern within the allocable memory block. *(Section 0075, lines 67-71 – Claim a "memwrite" function (which copies data into a specified area), a "memcheck" function (which checks the memory using specified options, and a "memlist" function (which detects a list of allocated memory blocks))*

Claim 7. The method of claim 6, wherein the memory overwrite detection pattern is written within an area of the allocable memory block that is used for alignment purposes. *(Section 0075, lines 67-68 – Claim a "memwrite" function which copies "data of a given size to a specified block". This allows for alignment)*

Claim 8. The method of claim 1, wherein an identifier is generated and stored for a routine that requests the allocable memory block and *(Section 0038, lines 11-18 – State the following: "The memory manager returns to the requesting process a heap handle which in a preferred embodiment contains, but is not limited to, the following information: 1) starting address of a memory heap or shared memory segment; 2) a flag containing a set of heap options for each bit that is set on and 3) a heap identifier. This heap handle is used for all memory requests involving memory pools including freeing the heap. ")*

- an identifier is generated and stored for a routine that frees the memory block. *(Section 0039, lines 15-22 – State the following: "a pool identifier which in a preferred embodiment is the unique user's pool offset value, for example an offset from the heap starting address to the beginning of a pool header without walls and any memory debug attachments, and 4) actual pool offset from the heap starting address. This pool handle is used for all memory requests involving memory blocks including freeing the pool")*

Claim 9.   The method of claim 1, further comprising storing a heap index for the allocable memory block within the allocable memory block, wherein the heap index points to one of a plurality of heaps. *(Section 0003, lines 3-6 – State the following: "The manager allocates a block of requested size from the heap and returns a handle or pointer to the block that is then typically used by the requesting program to access the block.")*

Claim 11.   A computer-readable medium having computer-executable components for tagging an allocable memory block, comprising: *(Abstract, lines 1-3 – State the following: "A data structure, method and system are provided incorporating a general purpose memory allocator and defensive heap memory manager. Figure 1, Number 424 – Shows a computer readable medium)*

- a first component that is arranged to determine the identity of a routine performing one of requesting the allocable memory block, requesting the size of the allocable memory block, and freeing the allocable memory block; *(Section 0039, lines 1-13 – State the following: "After receiving a request for allocating a memory pool, the memory manager performs all necessary size alignments and adjustments related to the specified memory debug mode (they are described below), allocates a new memory pool from available free heap space, creates and initializes a pool header (specific pool header data structures are described below), updates heap header data using a heap handle provided, and inserts a free block of the size of total pool space minus pool header size and memory debug data size into a skip list of free memory blocks. The heap memory manager returns to the requesting process a pool handle which in a preferred embodiment contains, but is not limited to, the following information...)*

- a second component that is arranged to generate an identifier for the routine; and *(Section 0039, lines 1-22 – Describe how the memory manager acts as an identifier generator)*

- a third component that is arranged to store the identifier in the allocable memory block. *(Section 0039, lines 1-22 – Describe how the memory manager stores the identifier in the allocable memory block)*

Claim 12.   The computer-readable medium of claim 11, further comprising an examination component that is arranged to examine the heap to determine the presence of memory errors. *(Section 0029, lines 8-22 – Describe how the heap is examined for memory errors)*

Claim 13.   The computer-readable medium of claim 12, further comprising a checksum component that is arranged to perform a checksum on the allocable memory block and *(Section 0066, lines 18-21 – State that there is a checksum done on an allocable memory block)*

- storing the results of the checksum within the allocable memory block. *(Section 0070, lines 15-32 – Show how the checksum is stored in a allocable memory block)*

Claim 14.   The computer-readable medium of claim 13, further comprising a checksum examination component that is arranged to examine the results of the checksum to determine the presence of memory errors. *(Section 0068, lines 1-7 – State that the checksum is examined to see if there are errors)*

Claim 15.   The computer-readable medium of claim 11, wherein the identifier is the return address of the identified routine. *(Section 0039, lines 16-19 – State the following: "a pool identifier which in a preferred embodiment is the unique user's pool offset value, for example an offset from the heap starting address to the beginning of a pool header without walls and any memory debug attachments")*

Claim 16.   The computer-readable medium of claim 11, further comprising a pattern component that is arranged to write a memory overwrite detection pattern within the allocable memory block. *(Section 0075, lines 67-71 – Claim a "memwrite" function (which copies data into a specified area), a "memcheck" function (which checks the memory using specified options,*

and a "memlist" function (which detects a list of allocated memory blocks))

Claim 17.   The computer-readable medium of claim 16, wherein the memory overwrite detection pattern is written within an area of the allocable memory block that is used for alignment purposes. *(Section 0075, lines 67-68 – Claim a "memwrite" function which copies "data of a given size to a specified block". This allows for alignment)*

Claim 18.   The computer-readable medium of claim 11, wherein an identifier is generated and stored for a routine that requests the allocable memory block and *(Section 0038, lines 11-18 – State the following: "The memory manager returns to the requesting process a heap handle which in a preferred embodiment contains, but is not limited to, the following information: 1) starting address of a memory heap or shared memory segment; 2) a flag containing a set of heap options for each bit that is set on and 3) a heap identifier. This heap handle is used for all memory requests involving memory pools including freeing the heap. ")*

- an identifier is generated and stored for a routine that frees the memory block. *(Section 0039, lines 15-22 – State the following: "a pool identifier which in a preferred embodiment is the unique user's pool offset value, for example an offset from the heap starting address to the beginning of a pool header without walls and any memory debug attachments, and 4) actual pool offset from the heap starting address. This pool handle is used for all memory requests involving memory blocks including freeing the pool")*

Claim 19.   The computer-readable medium of claim 11, further comprising an indexing component that is arranged to store a heap index for the allocable memory block within the allocable memory block, wherein the heap index points to one of a plurality of heaps. *(Section 0003, lines 3-6 – State the following: "The manager allocates a block of requested size from the heap*

and returns a handle or pointer to the block that is then typically used by the requesting program to access the block.")

Claim 21.   A system for tagging an allocable memory block, comprising: *(Abstract, lines 1-3 – State the following: "A data structure, method and system are provided incorporating a general purpose memory allocator and defensive heap memory manager)*

- a computer memory that comprises a heap in which allocable memory blocks can be allocated and freed; *(Abstract, lines 11-14 – Declare a heap memory)*

- a routine identifier that is arranged to determine the identity of a routine performing one of requesting the allocable memory block, requesting the size of the allocable memory block, and freeing the allocable memory block; *(Section 0039, lines 1-13 – State the following: "After receiving a request for allocating a memory pool, the memory manager performs all necessary size alignments and adjustments related to the specified memory debug mode (they are described below), allocates a new memory pool from available free heap space, creates and initializes a pool header (specific pool header data structures are described below), updates heap header data using a heap handle provided, and inserts a free block of the size of total pool space minus pool header size and memory debug data size into a skip list of free memory blocks. The heap memory manager returns to the requesting process a pool handle which in a preferred embodiment contains, but is not limited to, the following information...)*

- an identifier generator that is arranged to generate an identifier for the routine; and *(Section 0039, lines 1-22 – Describe how the memory manager acts as an identifier generator)*

- a diagnostic tagger that is arranged to store the identifier in the allocable memory block. *(Section 0039, lines 1-22 – Describe how the memory manager stores the identifier in the allocable memory block)*

Claim 22.   The system of claim 21, further comprising a memory verification system
that is arranged to examine the heap to determine the presence of memory
errors. *(Section 0029, lines 8-22 – Describe how the heap is examined for
memory errors)*

Claim 23.   The system of claim 22, further comprising a memory verification system
that is arranged to perform a checksum on the allocable memory block and
*(Section 0066, lines 18-21 – State that there is a checksum done on an
allocable memory block)*

-    storing the results of the checksum within the allocable memory block.
     *(Section 0070, lines 15-32 – Show how the checksum is stored in a
     allocable memory block)*

Claim 24.   The system of claim 23, further comprising a memory verification system
that is arranged to examine the results of the checksum to determine the
presence of memory errors. *(Section 0068, lines 1-7 – State that the
checksum is examined to see if there are errors)*

Claim 25.   The system of claim 21, wherein the identifier is the return address of the
identified routine. *(Section 0039, lines 16-19 – State the following: "a
pool identifier which in a preferred embodiment is the unique user's pool
offset value, for example an offset from the heap starting address to the
beginning of a pool header without walls and any memory debug
attachments")*

Claim 26.   The system of claim 21, further comprising a memory verification system
that is arranged to write a memory overwrite detection pattern within the
allocable memory block. *(Section 0075, lines 67-71 – Claim a
"memwrite" function (which copies data into a specified area), a
"memcheck" function (which checks the memory using specified options,
and a "memlist" function (which detects a list of allocated memory
blocks))*

Claim 27.   The system of claim 26, wherein the memory overwrite detection pattern
is written within an area of the allocable memory block that is used for

alignment purposes. *(Section 0075, lines 67-68 – Claim a "memwrite" function which copies "data of a given size to a specified block". This allows for alignment)*

Claim 28.   The system of claim 21, wherein an identifier is generated and stored for a routine that requests the allocable memory block and *(Section 0038, lines 11-18 – State the following: "The memory manager returns to the requesting process a heap handle which in a preferred embodiment contains, but is not limited to, the following information: 1) starting address of a memory heap or shared memory segment; 2) a flag containing a set of heap options for each bit that is set on and 3) a heap identifier. This heap handle is used for all memory requests involving memory pools including freeing the heap. ")*

- an identifier is generated and stored for a routine that frees the memory block. *(Section 0039, lines 15-22 – State the following: "a pool identifier which in a preferred embodiment is the unique user's pool offset value, for example an offset from the heap starting address to the beginning of a pool header without walls and any memory debug attachments, and 4) actual pool offset from the heap starting address. This pool handle is used for all memory requests involving memory blocks including freeing the pool")*

Claim 29.   The system of claim 21, further comprising a memory indexing system that is arranged to store a heap index for the allocable memory block within the allocable memory block, wherein the heap index points to one of a plurality of heaps. *(Section 0003, lines 3-6 – State the following: "The manager allocates a block of requested size from the heap and returns a handle or pointer to the block that is then typically used by the requesting program to access the block. ")*

Claim Rejections - 35 USC § 103

3.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

4.   Claims 10, 20, and 30 are rejected under 35 U.S.C.103(a) as being unpatentable over

Abrashkevich et al. as applied to claims 1, 11, and 21 above, further in view of Noel et al. (US

Patent 6,381,682).

Abrashkevich teaches the limitations of claims 1, 11 and 21 for the reasons above.

Abrashkevich 's invention differs from the claimed invention in that there is no specific
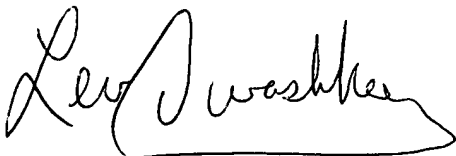
reference to a timestamp.

Abrashkevich fails to teach claims 10, 20, and 30, which all state that the invention is

"further comprising a memory timestamp system that is arranged to store a timestamp within the

allocable memory block, wherein the timestamp indicates the time when one of requesting and

freeing the allocable memory block is performed". However, Noel declares a "timestamp"

(Column 24, line 31) in her apparatus. Therefore, it would have been obvious to one of ordinary

skill in the art to combine the "Defensive Heap Memory Management" of Abrashkevich with

Noel's "Method and Apparatus for Dynamically Sharing Memory in a Processor System" so that

a timestamp would be included, so that the time of memory block allocation could be indicated

in order to keep the system as efficient and accurate as possible.

Conclusion

5.   Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Lev I. Iwashko whose telephone number is (571)272-1658. The

examiner can normally be reached on M-F (alternating Fridays), from 8-4PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Matt Kim can be reached on (571)272-4182. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for published

applications may be obtained from either Private PAIR or Public PAIR. Status information for

unpublished applications is available through Private PAIR only. For more information about

the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the

Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Lev Iwashko

MATTHEW KIM
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100